5. Ina

Program Name: Ina.java **Input File: ina.dat**

The Universal Internet Lab (UIL) recently developed an ingenious new method of connecting its computers. Each computer is directly connected to some other set of computers such that there is exactly one sequence of connections between any pair of computers. Stated differently, by treating the computers as vertices and connections as edges, the computers and their connections form a tree.

While this may seem constraining, connecting computers in this manner has some benefits. For example, software updates can now be installed in a distributed manner. Instead of each computer connecting to some central server to download a software update, only one computer needs to be updated. Then, by using these connections between computers, one computer can update all the other computers directly or indirectly!

Here's how it works: consider a connection where one end of the connection has a computer with the update (hereafter referred to as the source) and a computer without the update (hereafter referred to as the sink). Then in L seconds, the sink can download and install the update from the source. Each sink can only download from a single source at a time, and each source can only be accessed by a single sink at a time.

At the beginning, only computer S has the update. If Ina chooses the optimal ordering for updates, what is the fastest that all computers can be updated?

Input: The first line of input is an integer T (1 <= T <= 50), the number of test cases. Each test case starts with three space-separated integers N (1 <= N <= 100), S (1 <= S <= N) and L (1 <= L <= 1,000,000), the number of computers, the computer which initially has the update, and the amount of time it takes to transfer a single update. Then follow N - 1 lines, each with two space-separated integers U V (1 <= U, V <= N): the two computers connected by that connection. It is guaranteed that the set of edges forms a tree.

Output: For each test case, output a single integer: the minimum amount of time needed to update all computers. Format the output as in the samples.

Sample Description: In the first test case, an optimal solution is:

Computer 3 downloads from computer 1 (2 seconds)
Computer 4 downloads from computer 3 (2 seconds)
Simultaneously, computer 5 downloads from computer 3 and computer 2 downloads from computer 1 (2 seconds).

Thus the whole process takes 6 seconds. The final two updates can be done simultaneously because no computer is involved in multiple updates, and in each update one computer has the update and one does not.
(Sample input and output on next page)

Ina (cont.)

Sample input:

```
3
5 1 2
1 2
1 3
3 4
3 5
3 2 4
1 2
3 2
5 5 3
5 3
2 1
1 3
2 4
```

Sample output:

```
Case #1: 6
Case #2: 8
Case #3: 12
```